REMARKS/ARGUMENTS

Claims 1-37 are pending in the application. Claims 1, 19, 32 and 37 are amended. The undersigned thanks the Examiner for the telephone Interview on July 1, 2004.

Claims 1-14, 19-27, 32-35, and 37 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Cline et al. (U.S. 5,313,616); claims 15, 28 and 36 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Cline in view of Beizer, "Software Testing Techniques," 1986; and claims 16-18 and 29-31 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Cline. Applicants submit that all of the pending claims are patentable over the cited references, and reconsideration and allowance of these claims are respectfully requested.

The independent claims 1, 32, and 37 include, among other limitations, "parsing a source code of the computer program to identify behavior of an external function called by a function under test in the source code;" "generating a new function to be called by the function under test in the source code for mimicking some of the identified behavior of the external function;" "instrumenting the parsed source code by replacing the external function with the generated new function;" "compiling the instrumented code;" and "testing the compiled code, wherein the function under test calls the generated new function."

Cline does not teach or suggest "parsing a source code of the computer program to identify behavior of an external function called by a function under test in the source code." Rather, Cline describes generating (what he calls) "stubs" for

calling the code to certify "that tested and verified application programs will run on any hardware and operating systems which were designed in conformance with a set of system rules. "First, a conformance database is developed which includes allowable external calls, such as system calls and procedure calls. Then a static analysis is performed which analyzes the object code of an application program to determine whether any illegal or erroneous external calls are being made." (Col. 3, lines 7-13, emphasis added). This is accomplished by comparing the calls in the object code to the database of allowable external call. "Finally, a dynamic analysis of the program is performed which analyzes the application program as it is being run to determine any runtime errors in the calls made by the application program. If no errors are detected in either the static analysis or the dynamic analysis then the application program is certified to be compatible with the set of system rules [predetermined and stored in the database] and is transportable without change between all certified compatible computer systems." (Col. 3, lines 13-21, emphasis added).

For example, Cline emphasizes that "in a first step 60 the DBV inserts monitoring code into the application program's executable binary code. The application program is then exercised in a test harness in a step 62 so that the monitoring code can monitor and record system and procedure calls in a log database. Finally, a post-execution program called POST reads the log data base and calculates coverage statistics of the basic blocks and of the system calls, system procedure calls,

user procedure calls in a step 64." (Col. 15, lines 50-59, underlining added).

Therefore, the "stubs" of Cline are simply "monitoring code" that call the object code to collect statistic and compare the collected statistics to a log database to determine coverage. There is nothing in Cline that suggests identifying "behavior of an external function called by a function under test." Monitoring and recording system and procedure calls in a log database of Cline does not amount to identifying "behavior of an external function called by a function under test." Also, as mentioned before, Cline does not operate on a source code. Rather, it operates on an object code, in contrast to the claimed invention.

Additionally, Cline does not teach or suggest "generating a new function to be called by the function under test in the source code for mimicking some of the identified behavior of the external function," as recited by independent claims 1, 32, and 37. As discussed above, the monitoring code of Cline simply calls the object code to collect statistics and compares the collected statistics to a log database. In fact, a "System Procedure" stub and a "User Procedure" stub written in the 88000 assembly language are shown in TABLES 5 and 6 of Cline, respectively. (Col. 16). There is nothing in these tables that suggest "mimicking some of the identified behavior of the external function."

In fact, by emphasizing (in Col. 16, lines 29-34) that "DBV then scans the input source text for procedure calls (recognized by the bsr or bsr.n instructions) and makes a list of all of the

calls by location and target in a step 82. As a call target is listed, a 'stub' for that call is also generated and the current total stub size is tallied," Cline teaches away from "mimicking some of the identified behavior of the external function," as recited by the independent claims 1, 32, and 37.

Moreover, Cline does not teach or suggest "instrumenting the parsed source code by replacing the external function with the generated new function." Cline does not replace any external functions, rather, it adds (inserts) "monitoring code to the application program and verifying system and procedure calls and determining program coverage as the application program executes a test program. The monitoring code creates a log database which is used by a separate "post" program to print a report of call usage and program coverage after the completion of the application program run." (Col. 3, lines 29-37, also step 60 in Fig. 12, and col. 15, lines 50-52). In deed, by adding monitoring code, Cline teaches away from "replacing the external function with the generated new function," as recited by the independent claims 1, 32, and 37.

In addition, Cline does not teach or suggest "compiling the instrumented code." As discussed above, Cline operates on the object code and adds monitoring code to the object code. Therefore, the instrumented object code (application program, as Cline sometimes calls it) is already compiled.

Finally, Cline does not teach or suggest "wherein the function under test calls the generated new function." Rather, the monitoring code of Cline is "inserted" into the application program's executable code. (Col. 15, lines 50-52). Cline

emphasizes that "[t]he insertion of the monitoring code can be compared to a computer virus in that the operation of the inserted monitoring code merges with the operation of the application program and tends to run invisibly in the background. The monitoring code differs, however, from computer viruses in that it does not reproduce itself by injecting copies of itself into other programs. The monitoring code can be inserted into the application program as illustrated in FIG. 13. First, a hash table is built from the conformance database in a step 66 as previously described. Next, the application program is analyzed and processed in a step 68. Finally, in a step 70 the application program is assembled with portions of monitoring code and data to form the instrumented application program." (Col. 15, line 66 to col. 16, line 5). Therefore in Cline, the monitoring code merges and becomes part of the application program (under test), rather than being called by the function under test.

Accordingly, for at least the above-discussed five distinct claimed elements, independent claims 1, 32, and 37 are patentable over Cline.

Independent claim 19 includes, among other limitations, "parsing the source code of the computer program to identify behavior of a plurality of smaller components in the source code;" "generating stubs to emulate some of the identified behavior of plurality of smaller components;" "replacing one or more of the plurality of smaller components with one or more of respective generated stubs;" and "wherein the generated stubs

have the same signature as the respective replaced one or more of the plurality of smaller components."

As explained above, Cline does not parse the source code and does not identify the behavior of any components. Rather, it simply adds monitoring code to monitor the application program object code. Furthermore, Cline does not teach or suggest "generating stubs to emulate some of the identified behavior of plurality of smaller components." The stubs (monitoring code) of Cline do not "emulate some of the identified behavior of plurality of smaller components." They simply "monitor and record system and procedure calls in a log database."

Additionally, Cline does not teach or suggest "replacing one or more of the plurality of smaller components with one or more of respective generated stubs." As discussed above, Cline adds "monitoring code to the application program and verifying system and procedure calls." In fact by adding monitoring code, Cline teaches away from "replacing one or more of the plurality of smaller components with one or more of respective generated stubs," as recited by the independent claim 19.

Finally, Cline does not teach or suggest "wherein the generated stubs have the same signature as the respective replaced one or more of the plurality of smaller components." First, there is no disclosure in Cline that the monitoring code has the same signature as the application program. Second, as discussed previously, there is no component in application program of Cline that is replaced.

Accordingly, for at least the above-discussed four distinct claimed elements, independent claim 19 is patentable over Cline.

As a result, neither Cline, nor Beizer, alone or in combination, teach or suggest the above-mentioned limitations, required by independent claims 1, 19, 32 and 37. Dependent claims 2-18, 20-31 and 34-36 all depend, directly or indirectly from their respective independent claims. Therefore, these claims are also patentable over the cited references, as being dependent from allowable independent claims and for the additional limitations they include therein.

In view of the foregoing amendments and remarks, it is respectfully submitted that this application is now in condition for allowance, and accordingly, reconsideration and allowance are respectfully requested.

Respectfully submitted,
CHRISTIE, PARKER & HALE, LLP

By _____
Raymond R. Tabandeh
Reg. No. 43,945
626/795-9900

RRT/clv
CLV PAS572948.1-*-07/2/04 3:32 PM